(NASA-CR-175756) EXPERIMENTAL AND ANALYTICAL INVESTIGATION OF FAN FLOW INTERACTION WITH DOWNSTREAM STRUTS Interim Report, 16 Jun. 1984 - 15 Mar. 1985 (Virginia Polytechnic Inst. and State Univ.) G3/02 15263
N85-25201
HC A03/MF A01
Unclas

COLLEGE OF ENGINEERING

VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY

BLACKSBURG, VIRGINIA

Interim Report on the Experimental and Analytical Investigation of Fan Flow Interaction with Downstream Struts

T. L. Olsen
W. F. Ng
W. F. O'Brien, Jr.

Mechanical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061

Interim Report #1
for the period June 16, 1984 to March 15, 1985

Prepared for:

National Aeronautics and Space Administration
NASA Grant Number NAG-1-486
James A. Schoenster, Technical Monitor
NASA-Langley Research Center

This is an informal interim report recommended at this time for internal use only. The contents will be included in a formal report suitable for public release or distribution at a later time.

April 1985

TABLE OF CONTENTS

I. Introduction and Research Objectives

An important class of turbomachinery noise is produced by the interaction of the blades of a machine with flow perturbations within the machine. Such flow perturbations may be stationary in time and position, as would result from the potential field of a downstream strut. Disturbances originating from struts exist continually during operation of a machine, and will contribute to tone noise depending on the strength of the consequent flow response and pressure fluctuations.

The VPI&SU Turbomachinery Research Group had for some years carried on an investigation which was designed to provide insight into the fundamental aspects of fan rotor-downstream strut interaction (1,2,3). High response, miniature pressure transducers were embedded in the rotor blades of an experimental fan rig. Five downstream struts were placed at several downstream locations in the discharge flow annulus of the single-stage machine (Fig. 1). Significant interaction of the rotor blade surface pressures with the flow disturbance produced by the downstream struts was measured. Several numerical procedures for calculating the quasi-steady rotor response due to downstream flow obstructions were developed (3). A preliminary comparison of experimental and calculated fluctuating blade pressures on the rotor blades shows general agreement between the experimental and calculated values. Although progress has been made as a result of this effort, there are still areas in which further work is needed.

The objective of the program is to investigate the interaction of fan rotor flow with downstream struts. The results of this investigation will be an improved understanding of noise prediction with the goal of developing a design-for-noise capability and development of a data base for use in validating analytical prediction methods and noise-reduction concepts. The experimental program will focus on further data reduction of the existing data base from previous work.

The signal from the blade mounted pressure transducers will be used to calculate the unsteady lift and moment coefficients. In addition, power spectral analysis on the time-averaged data will be performed to retain the phase information. Both of these analyses will be used to validate the time-marching code, which was written to obtain the rotor blade flow response to a downstream strut row.

The present report gives a summary of the work for the period 6/16/84 - 3/15/85. Major topics to be covered in this report include the data transfer from NASA Langley Research Center to VPI&SU, further data reduction and the calculation of unsteady lift and moment. The report will conclude with a discussion of future plans.

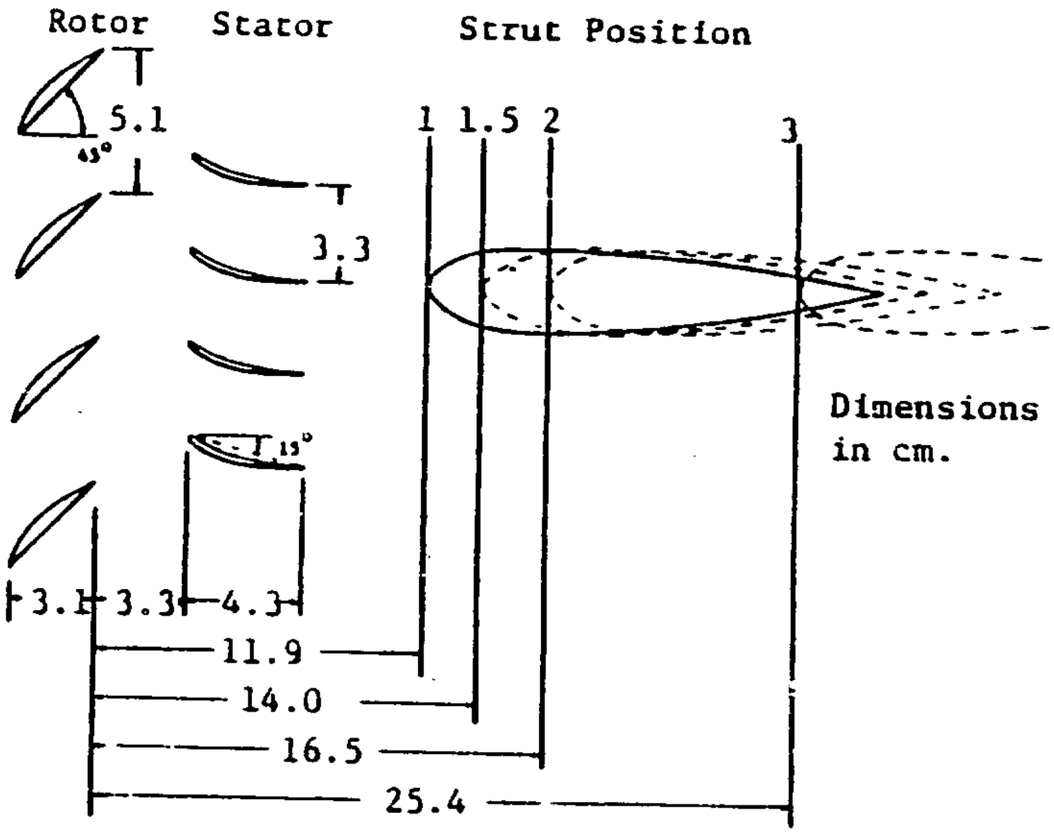

Fig. 1 Cross section of compressor showing rotor-stator-strut spacing

II. Data Transfer NASA Langley Research Center to VPI&SU

This section describes in detail the transfer of analog data recorded using FM tape recorder to digital data that can be reduced using computers at VPI&SU. A listing of the programs is provided in the Appendix.

1. Digitization of Analog Data

The original analog data were recorded on half-inch tape running at a speed of 15 inches per second (ips). The data were digitized in the Central Data Transcription Facility at NASA Langley Research Center, Analysis and Computation Division, Computer Management Branch, Data Management Section. The data were digitized with an anti-aliasing filter set at 5 kHz flat. The digitization gave 360 points per cycle of rotor revolution. Since the rotor rotational speed fluctuates with time, the corresponding data sample rate will have to be adjusted to account for this. The time increment between data point varies between 50 and 70 microseconds. The once per revolution signal is used to determine the sampling rate to give 360 points per rotation. The digitized data were written on magnetic tape using the UNIVAX VARIAN Model 77-600 computer. The specifications on the format of the data written on the magnetic tape include the following: Nine tracks; Non-labeled; Edcdic; 1600 BPI; Fixed block binary; 16 Bit Integer Words and Two's compliment (Integer *2). The characteristics of the digitized data file on magnetic tape include the following: Blocksize 3840, Record length 3840 and Record format F.

2. Transfer of digitized data to computer at VPI&SU

The digitized data is transferred to an IBM 3081 system. The system operates under the control of VM (Virtual Machine) and OS/VS 2,3 (MVS) and has twenty-four (24) megabytes of memory. A series of software programs were developed to facilitate the data transfer. Figure 2 shows the flow chart to read the data off the magnetic tape. The first program, TPP CNTL, executes a system program (TPPRINT) which checks the tape characteristics. The program TPPRINT scans a tape and tells the number of files, the block storage size, and the number of records per block (thus record length). Record length is equal to blocksize/(number of records per block). TPPRINT also gives a machine language translation of the stored data, which is not readily readable.

Next step in the data transfer is to copy NASA's tape to a VPI system tape. The VPI system tape is permanently active, easier to access and is in readable format. Program TAPE TAPE is written to create the system tape.

Two programs were written to read from the system tape. TAPE LISTE shows the first 20 elements of every line. This is not a necessary operation, but allows a visual look at a swath of the whole file from top to bottom. The second program, TAPE LISTL, shows the first few lines of the file in totality. This allows viewing the arrangement of data on each line. Some prior knowledge of the data expected aids in selecting the format used in TAPE LISTL. (See appendix for a sample output from the program TAPE LISTL).

As mentioned earlier, the time increment between data point varies between 50 and 70 microseconds. The rotor test speed of 2900 RPM fluc-

tuates. The analog data were digitized by 1) assigning a square wave pulse to match the once-per-rev signal, 2) dividing each pulse into 360 units, and 3) making a digital reading of each channel at each of 360 cycle positions in every revolution. Thus ΔTime $\equiv$ ΔPosition/rotor speed. Since the rotor speed fluctuates, the time increment fluctuates and thus cannot correlate exactly with position. It is decided that the data will be analyzed in the position domain.

Next, the square wave must be checked to see that 360 readings occur in each cycle. The file POS CHECK searches for the drop in each square wave (6000^{+} followed by 600^{+}) and counts the number of entries since the preceeding square wave. It then shows the number of cycle positions in each revolutions, some truncation may occur and some cycles show 359 or 361 positions instead of 360. The approach taken here is to discard the bad cycles and the program CYCLE SELECT is written to do that. The program also ignores the first incomplete cycle and provides a printout of data points retained. A similar file, VOLT PLOT, performs the same data manipulation and then plots each channel, the square wave, and the once-per-rev signal versus cycle position for three good cycles.

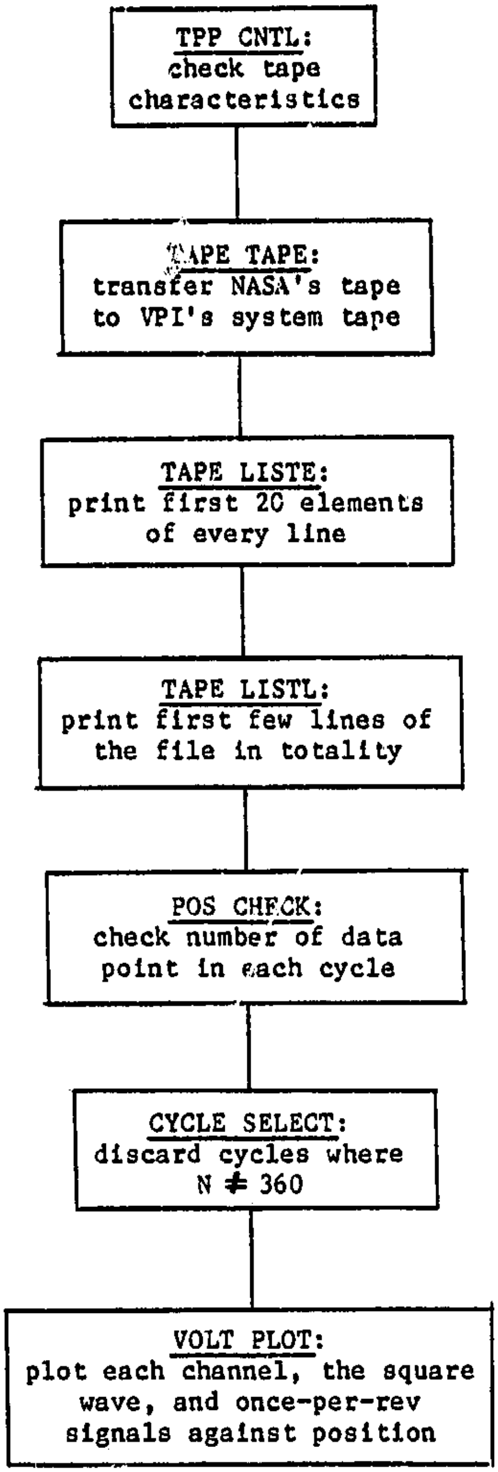

Fig. 2 Block diagram showing programs used in data transfer

III. Other Accomplishments

A. Presentation and Publication

"Calculation of Unsteady Fan Rotor Response Caused by Downstream Flow Distortions," by O'Brien, W. F., Jr., Richardson, S. M., and Ng, W. F., AIAA Paper No. 84-2282, presented by W. F. Ng at the AIAA 9th Aeroacoustics Conference, Williamsburg, Virginia, October 15-17, 1984, submitted for publication to AIAA *Journal of Propulsion and Power*.

B. Improvement in Analytical Code

Refinement in the time-marching code has led to improvement in the convergence of the solution. The calculation was carried out to 800 cycles with error in mass conservation less than 0.7% and error in total pressure conservation within 0.2%. The predicted pressure fluctuations on the blade surfaces compared very well with the measured data [4]. In addition, the Douglas-Newman program and the time-marching code are now better documented and readily available for future use.

C. Data Reduction

All analog data recorded on tapes had been digitized and transferred to the computers at VPI&SU for further data reduction. A computer program was written to take ensemble average on the data. Figure 3 shows some typical raw data and 200 cycles ensemble averaged data from pressure transducers mounted on the pressure and suction sides of the blade, with and without the downstream struts. The rotor is running at 2900 RPM and 15% surge margin. The transducers are at 50% span.

A program is also developed to calculate the unsteady lift and moment from the blade mounted transducers. The lift is calculated using

$$L = \sum_{i=1}^{6} (P_p - P_s)_i \, \Delta A_i \tag{1}$$

where P_p and P_s are the pressures on the pressure and suction surfaces of the blade, ΔA is the differential area covered by each transducer. the pressures can be written as two components:

$$P_p(t) = \overline{P_p} + P_p'(t)$$

and

$$P_s(t) = \overline{P_s} + P_s'(t)$$

where $\overline{P}$ is the steady-state pressure and P' is the time-varying pressure. Thus

$$L = \sum_{i=1}^{6} (\overline{P_p} - \overline{P_s})_i \, \Delta A_i + \sum_{i=1}^{6} (P_p' - P_s')_i \, \Delta A_i \tag{2}$$

The first term is a constant and does not change with time. The second term can be calculated from the measurement using the blade mounted transducers. Figure 4.A shows the unsteady lift against rotor position for different downstream strut locations.

Similarly the unsteady moment is given by

$$M = \sum_{i=1}^{6} d_i(\overline{P}_p - \overline{P}_s)\Delta A_i + \sum_{i=1}^{6} d_i(P'_p - P'_s)\Delta A_i \qquad (3)$$

where d_i is the distance between the transducer and quarter chord from the leading edge of the blade.

Again the first term does not change with time and the second term can be calculated from the blade mounted transducers. Figure 4-B shows the unsteady moment plotted against rotor position for different downstream strut locations.

D. Interactions with Industry

There is strong interest on the current work from the Aircraft Engine Business Group of General Electric Company at Cincinnati, Ohio. Discussions with their aeroacoustic design group have indicated that the problem is still of definite interest to them and to the aircraft manufacturer.They may use the analytical code for future investigations. Two specific areas of interest are: (1) Optimum positioning of the stators between the rotors and the struts to minimize unsteady interaction, (2) Effect of nonuniform circumferential spacing of the support struts on the fan-strut interaction (5). There will be continuous interactions between the Turbomachinery Research Group and aircraft engine industry.

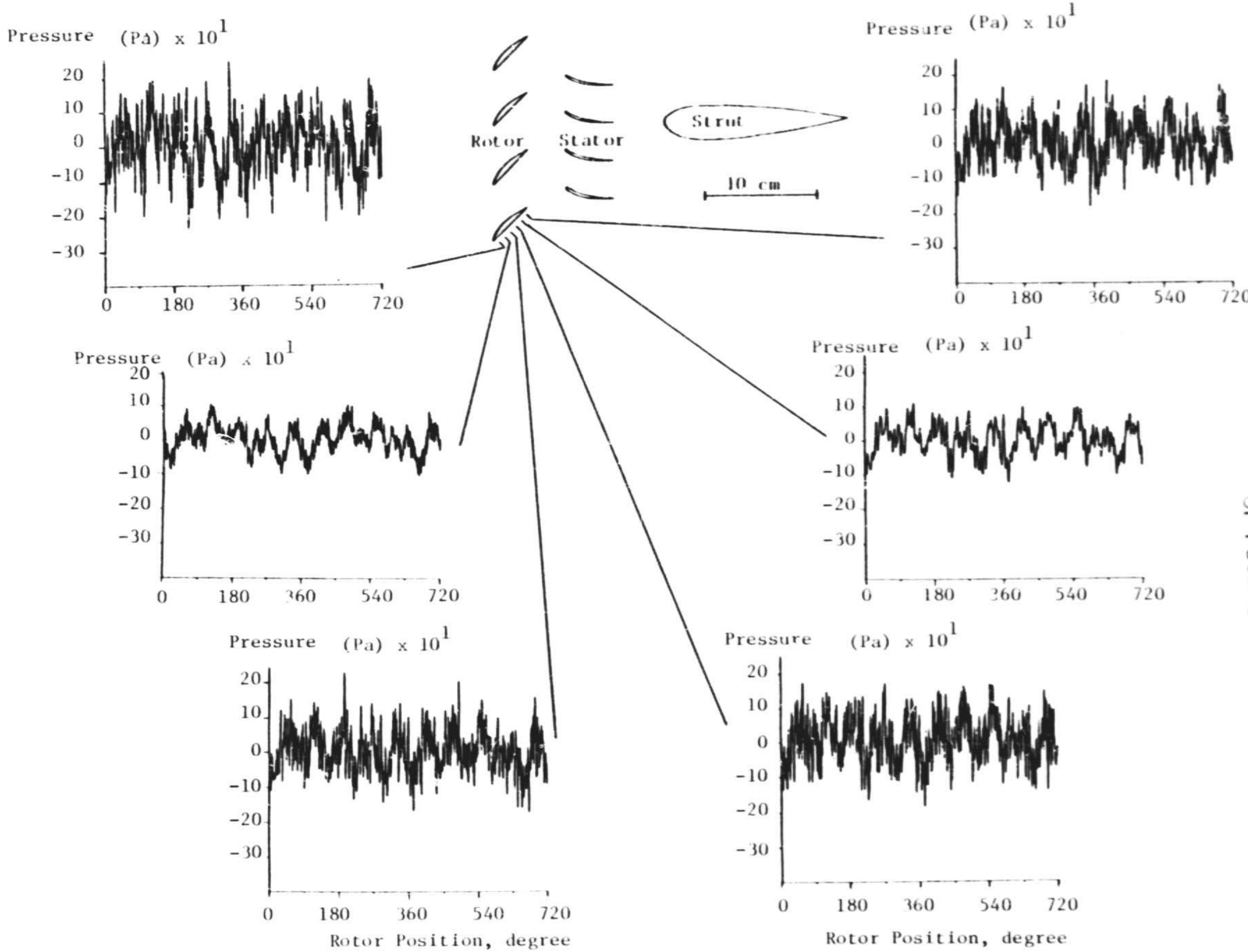

Fig. 3.A.1. Unsteady pressure on pressuresurface of rotor (raw data) with strut at 0.119 m from rotor trailing edge.

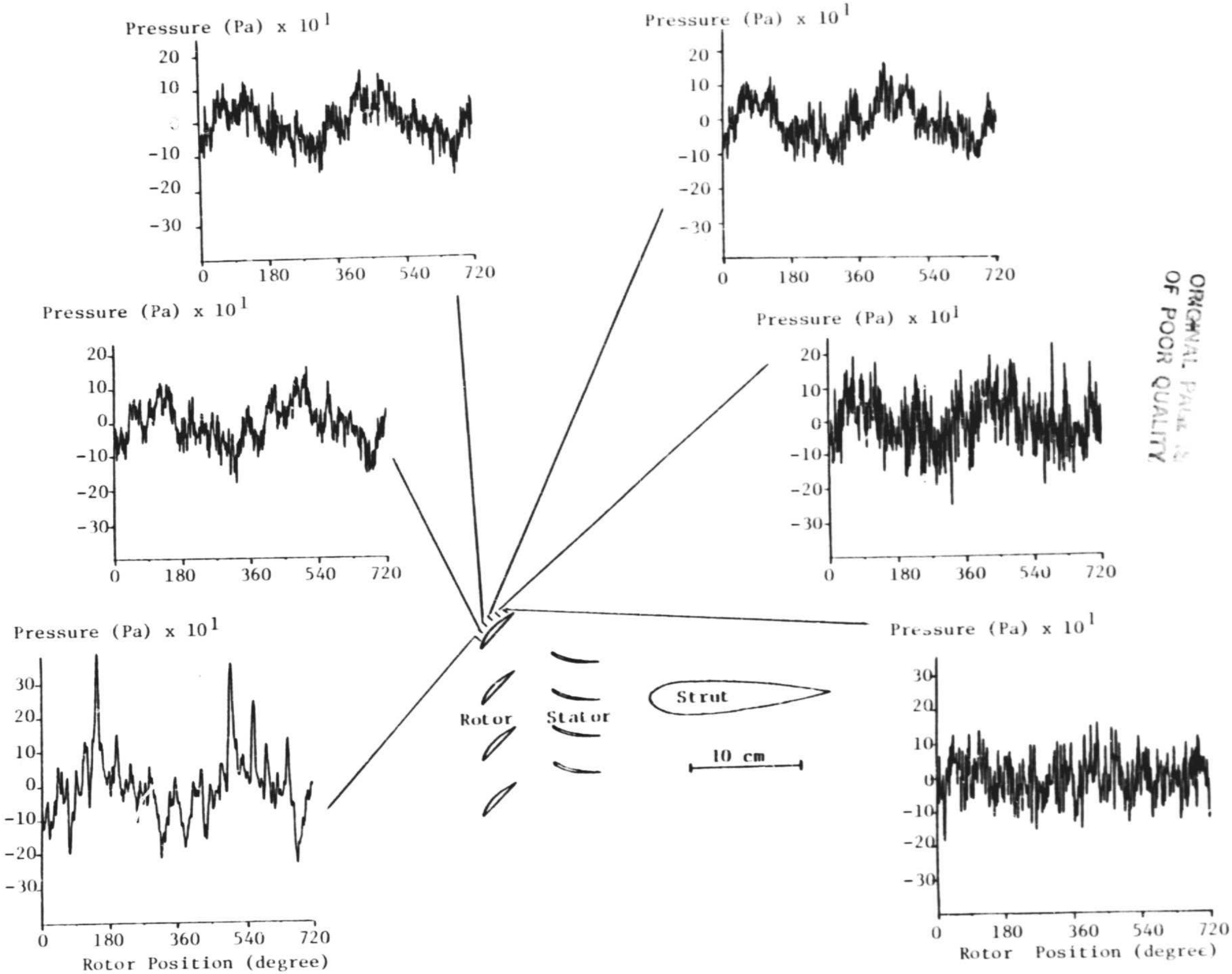

Fig. 3.A.2. Unsteady pressure on suction surface of rotor (raw data) with strut at 0.119 m from rotor trailing edge.

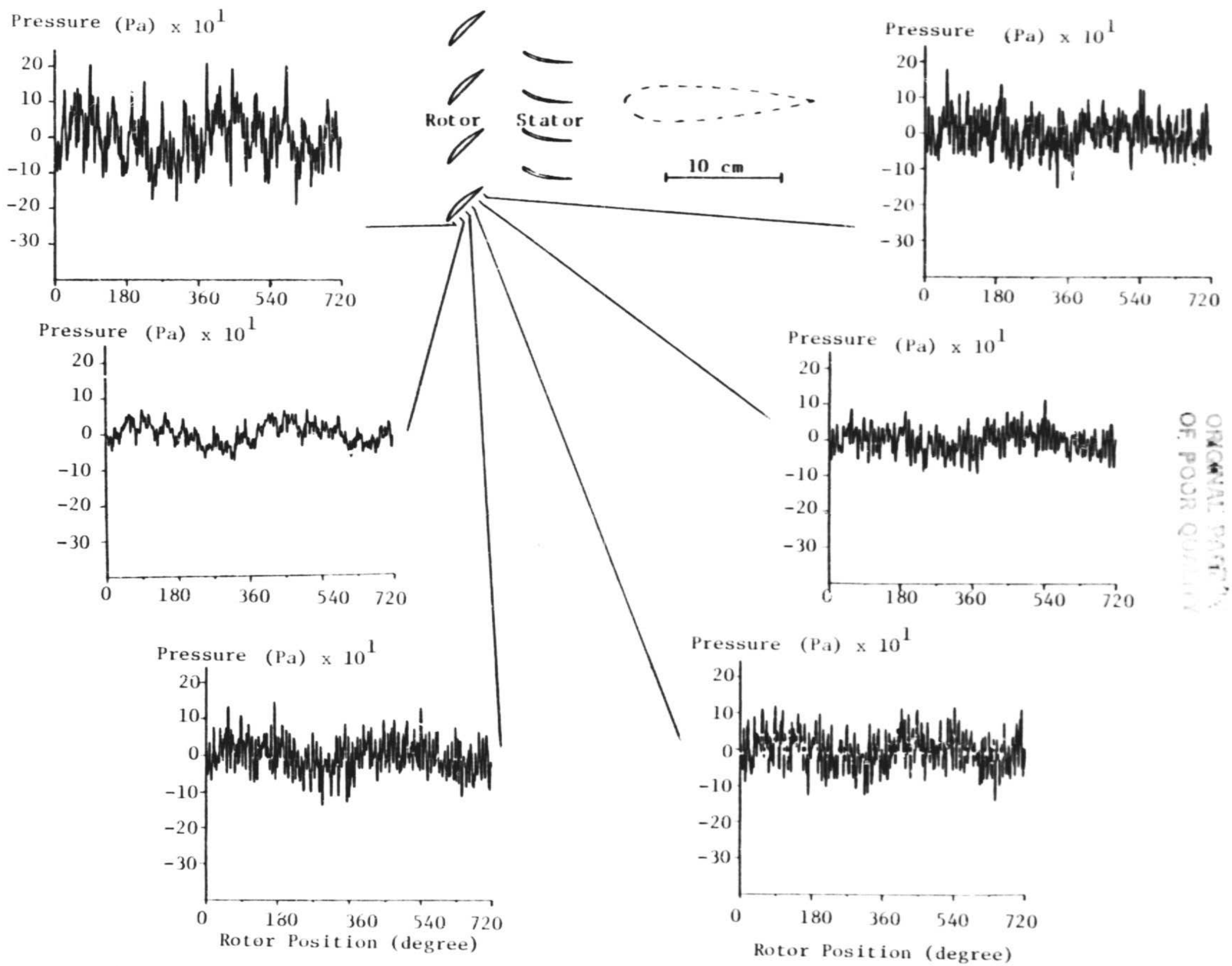

Fig. 3.B.1. Unsteady pressure on pressure surface of rotor (raw data) with strut removed.

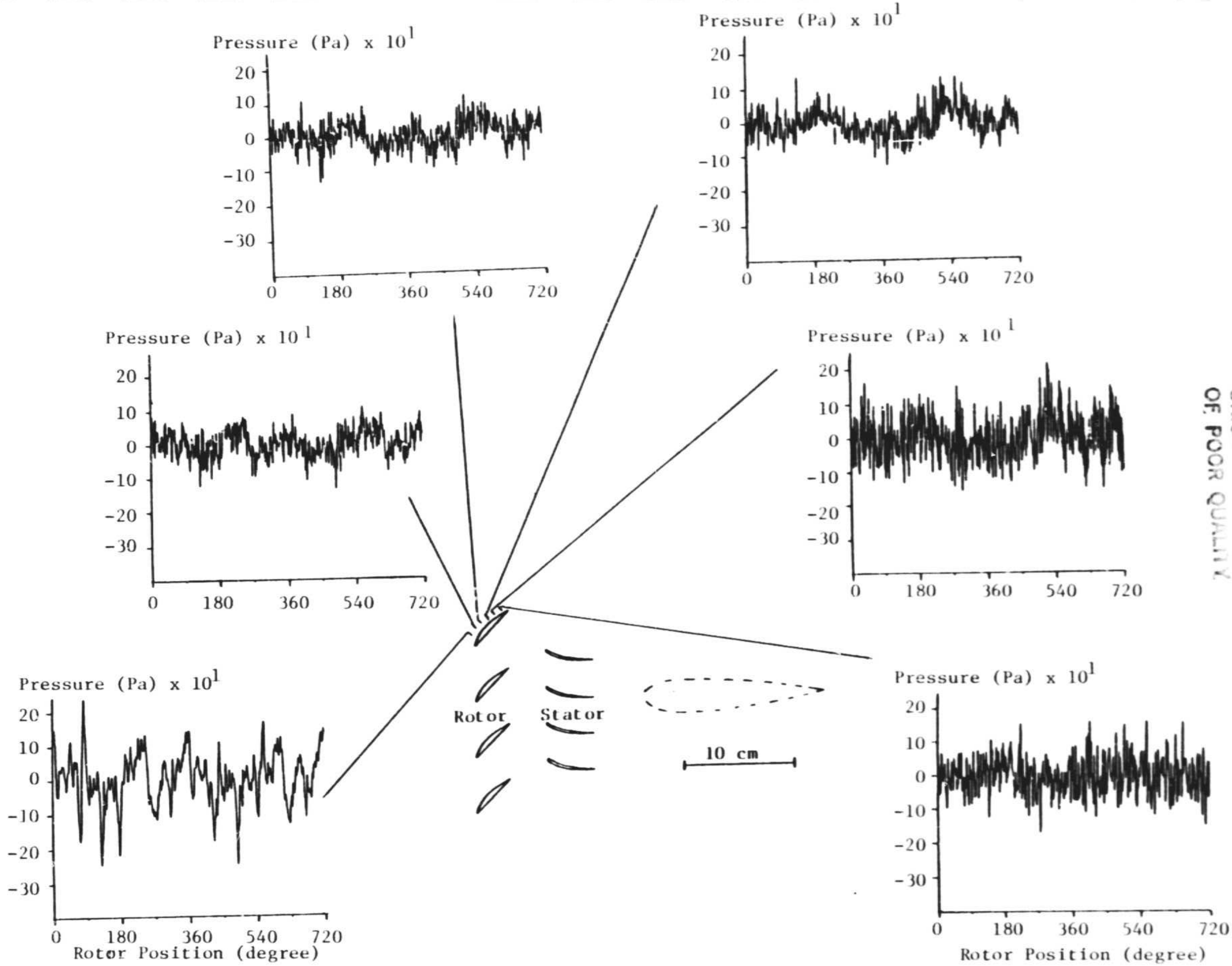

Fig. 3.B.2. Unsteady pressure on suction surface of rotor (raw data) with strut removed.

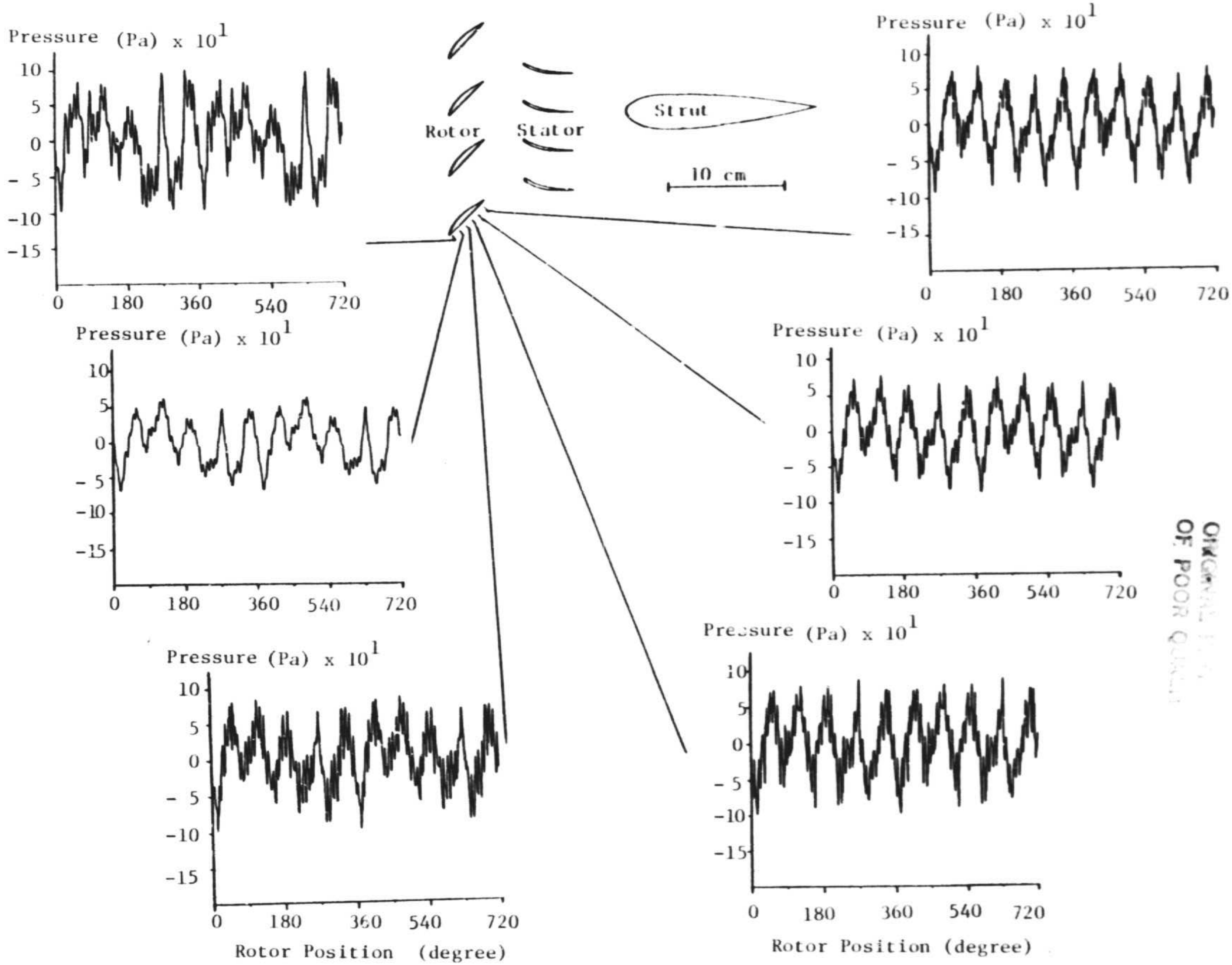

Fig. 3.C.1. Unsteady pressure on pressure surface of rotor (200 ensemble averaged) with strut at 0.119 m from rotor trailing edge.

ORIGINAL PAGE IS OF POOR QUALITY

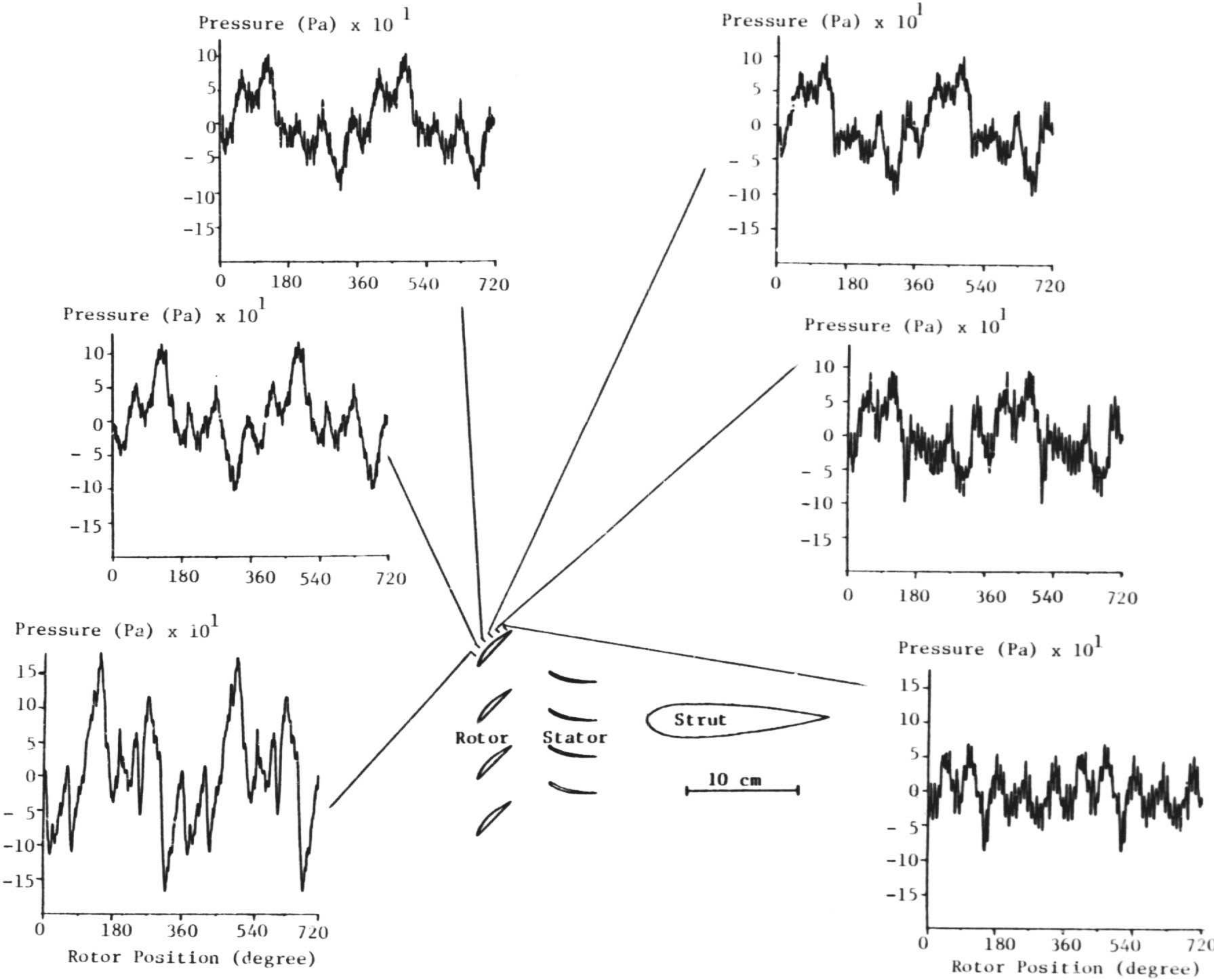

Fig. 3.C.2. Unsteady pressure on suction surface of rotor (200 ensemble averaged) with strut at 0.119 m from rotor trailing edge.

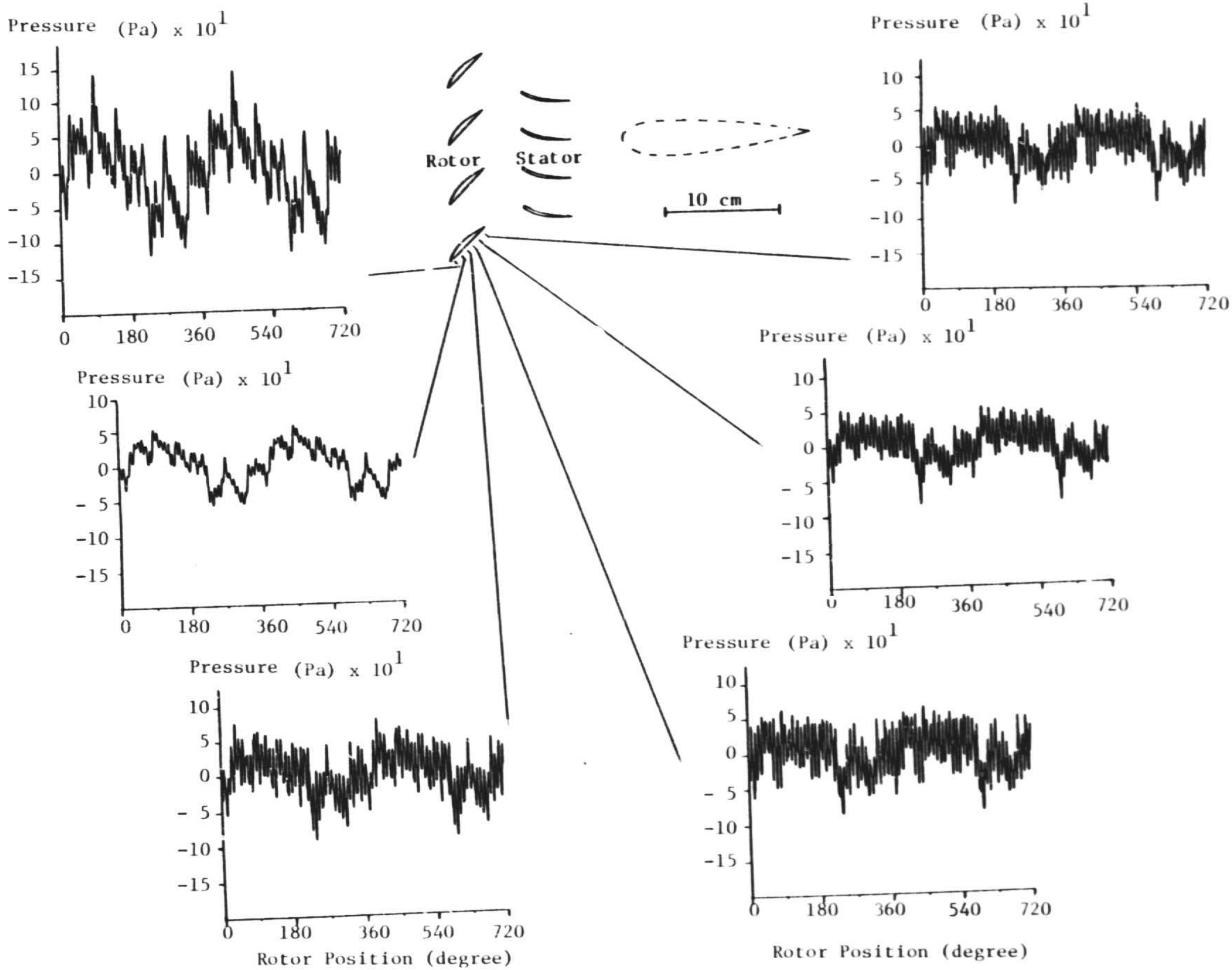

Fig. 3.D.1. Unsteady pressure on pressure surface of rotor (200 ensemble averaged) with strut removed.

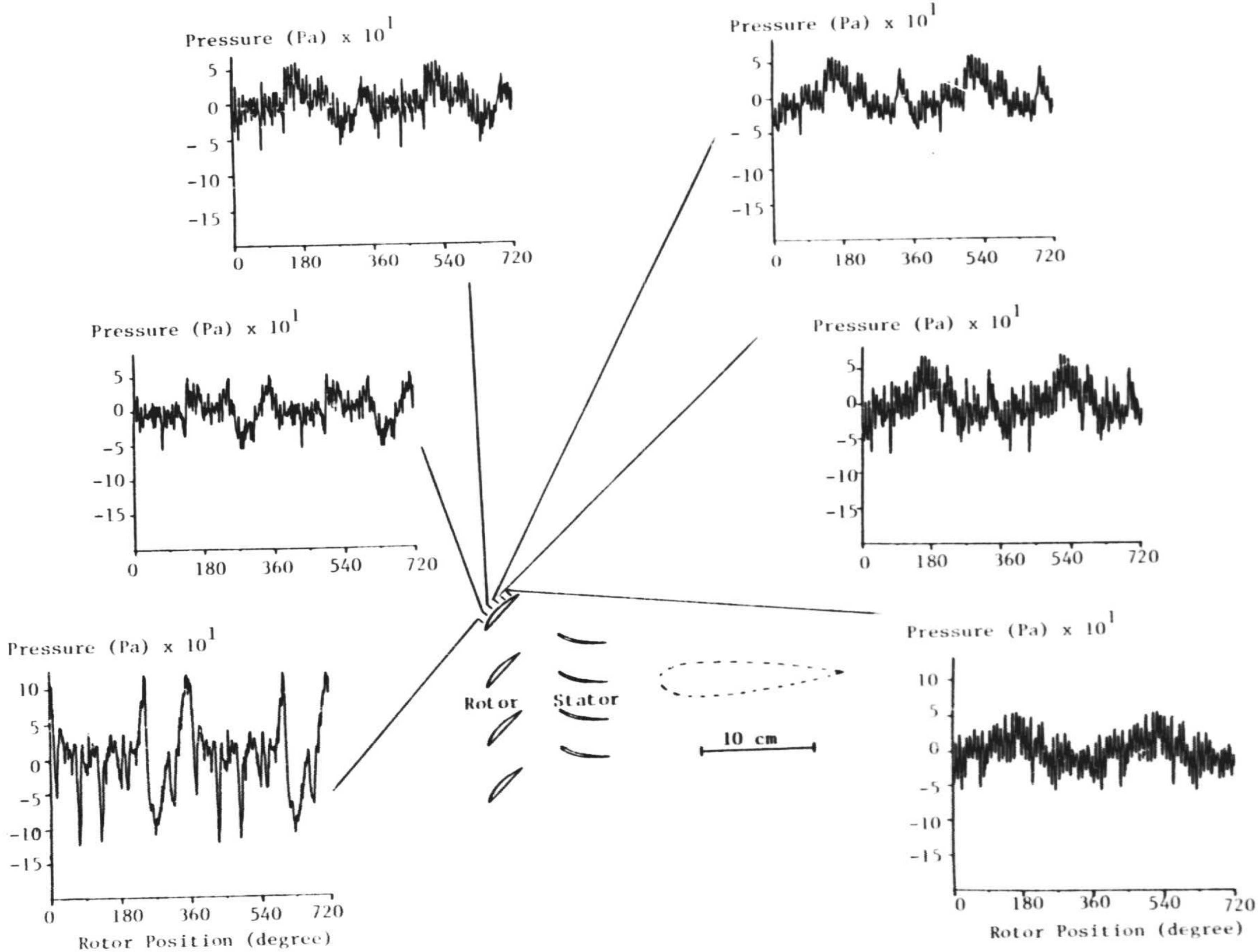

Fig. 3.D.2. Unsteady pressure on suction surface of rotor (200 ensemble averaged) with strut removed.

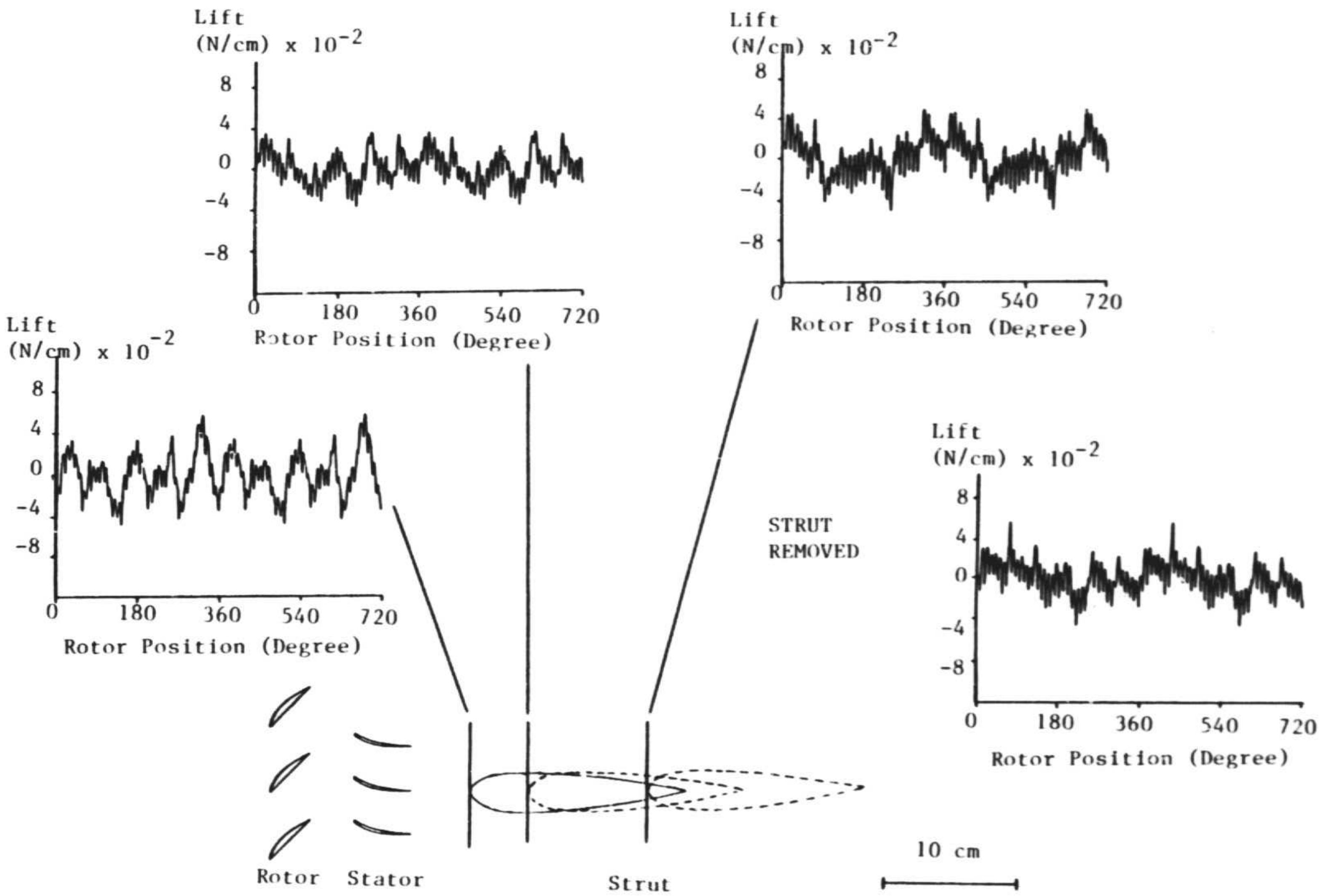

Fig. 4.A. Unsteady lift as a function of strut positions.

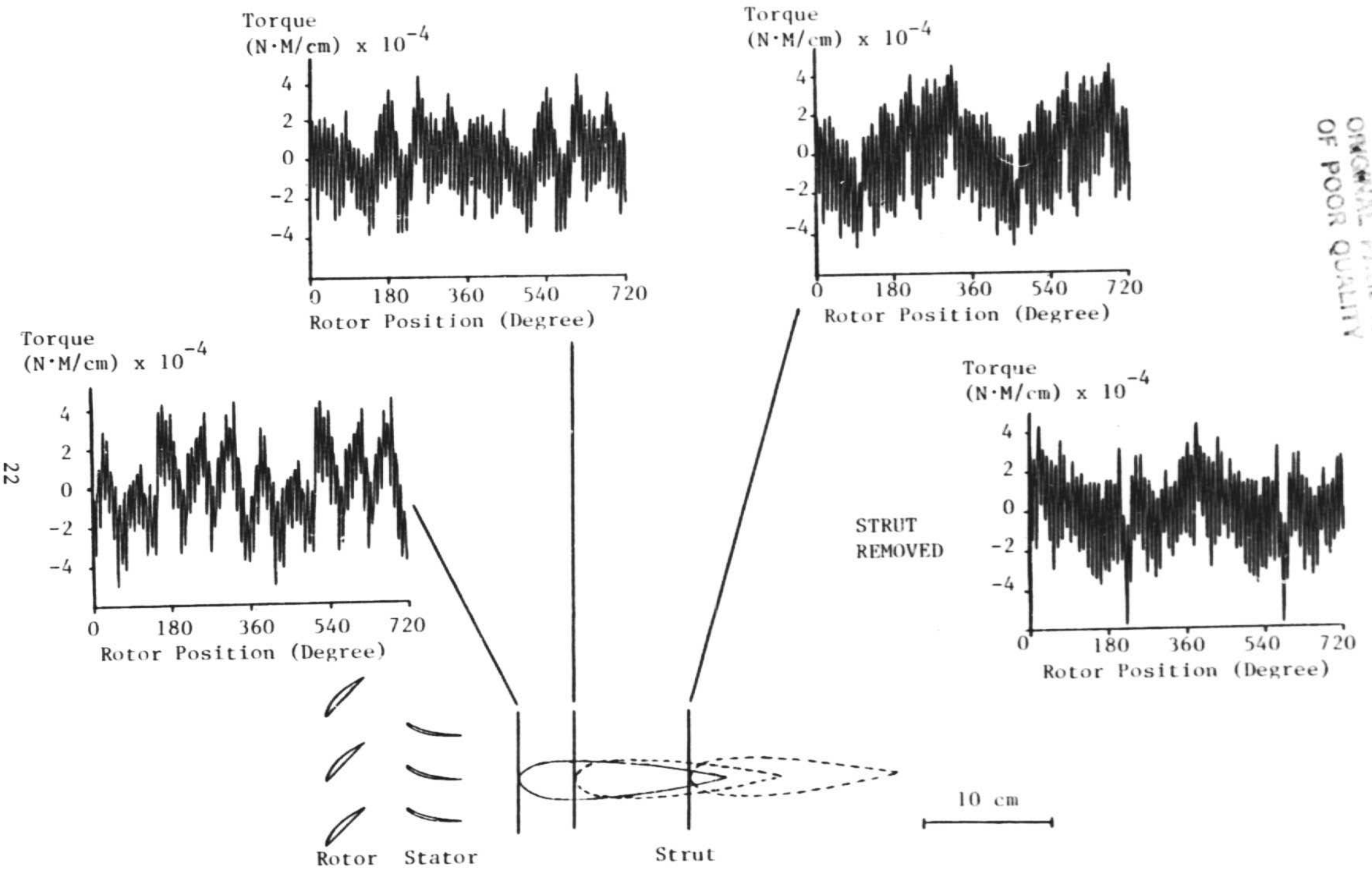

Fig. 4.B. Unsteady torque as a function of strut positions

IV. Summary and Future Plan

All analog data recorded on tapes have been digitized and successfully transferred to the computers at VPI&SU for further data reduction. In addition, a program was developed to calculate the unsteady lift and moment from the blade mounted transducers. The result of this analysis will be used to validate the numerical calculation. Work is in progress for implementing a digital Fast Fourier Transform (FFT) and Discrete Fourier Transform (DFT) subroutines for looking at spectral analysis of the data. A parametric study on the effect of different Fourier Transforms on the spectral analysis is underway.

In order to isolate the effect of downstream struts from other harmonic disturbances, it is necessary to subtract the amplitude of pressure fluctuation due to "background" disturbances from the amplitude of pressure fluctuation due to the downstream struts alone. In the previous data analysis, this was done by performing a power spectral analysis (FFT) on the raw data, then taking 200 averages of the power spectrum to give an overall spectral analysis. This was done for both cases; with and without the downstream struts. The difference in amplitude of the pressure fluctuation was then reported in Ref. (1). No phase information was retained in this data analysis.

Further data reduction will include an enhanced-signal spectral analysis (6) for the data set. The raw data will first be ensemble averaged (conditional averaging) and then a power spectral analysis will be performed on the averaged data while retaining the phase information. The vector difference in amplitude of the presure fluctuation (with and without the downstream struts) can be obtained using a polar

plot (Nyquist plot). The result of this analysis will be compared to previous results as well as to the analytical model.

In addition, using the existing analytical code, an effort will be initiated to investigate the optimum positioning of the stators between the rotors and the struts to minimize unsteady interaction.

V. References

1. O'Brien, W. F., Jr., Reimers, S. L., and Richardson, S. M., "Interaction of Fan Rotor Flow with Downstream Struts," AIAA-83-0682, 1983.

2. Reimers, S. L., "An Investigation of Fan Flow Interaction with a Downstream Strut," M.S. Thesis, Department of Mechanical Engineering, VPI&SU, 1983.

3. Richardson, S. M., "Numerical Procedures for Calculating the Unsteady Rotor Response Caused by Downstream Flow Distortions," M.S. Thesis, Department of Mechanical Engineering, VPI&SU, 1983.

4. O'Brien, W. F., Jr., Richardson, S. M., and Ng, W. F., "Calculation of Unsteady Fan Rotor Response Caused by Downstream Flow Distortions," AIAA-84-2282, 1984, also submitted for publication to AIAA Journal of Propulsion and Power.

5. Gliebe, P. R. and Ho, P. Y., personal communications, June, October 1984.

6. Schoenster, James A., "Fluctuating Pressures on Fan Blades of a Turbofan Engine," NASA TP-1976, March 1982.

Appendix: Listing of programs used in data transfer

1. Program TPP CNTL - execute a system program (TPPRINT) which checks the tape characteristics.

```
TPP      CNTL

//B0589TLO JOB 529E5,'TIM OLSEN',REGION=1024K,TIME=(,59)
/*JOBPARM LINES=99
/*PRIORITY STANDARD
/*LONGKEY WINGFAL
//STEP0001  EXEC PGM=TPPRINT     SYSTEM PROGRAM WHICH CHECKS TAPE
//SYSPRINT  DD SYSOUT=*          CHARACTERISTICS.
//SYSUT1    DD DSN=VPI13,
//             DISP=OLD,
//             UNIT=TAPE,
//             LABEL=(1,NL),
//             VOL=SER=VPI13,
//             DCB=BLKSIZE=32000 LARGEST POSSIBLE; TAPE WILL PROBABLY
//SYSUT2    DD SYSOUT=*          HAVE SMALLER BLOCKSIZE.  DSN AND VOL
//SYSIN     DD *                 GIVE NAME OF NASA TAPE BEING CHECKED.
FILES=99,STOPAFT=3
/*
//
```

2. Program TAPE TAPE - transfer NASA's tape to VPI's system tape.

```
TAPE     TAPE

//B0589TLO JOB 529E5,'TIM OLSEN',REGION=1536K,TIME=(,59)
/*LONGKEY WINGFAL
/*ROUTE PRINT VPIVM1.OLSENTL
/*ROUTE PUNCH VPIVM1.OLSENTL
/*JOBPARM LINES=10                          'DATA' DIMENSION IS HALF OF
/*PRIORITY IDLE                             THE TAPE'S RECORD LENGTH.
//STEP0001 EXEC PLIXCLG                     CALL PL1 COMPILER.
//PLI.SYSIN DD *                            PL1 PROGRAM:
* PROCESS SOURCE, FLAG(I), GOSTMT ;
 READTP :
    PROCEDURE OPTIONS( MAIN ) ;
    DECLARE SYSPRINT OUTPUT PRINT FILE ;
    DECLARE SYSIN INPUT RECORD FILE ;
    DECLARE OUTFIL OUTPUT RECORD FILE ;
    DECLARE DATA( 1920 ) FIXED BINARY( 15 ) ;
    DECLARE COUNT FIXED BINARY( 31 ) ;
    DECLARE END_OF_FILE FIXED BINARY( 31 ) INITIAL( 0 ) ;
    ON ENDFILE( SYSIN ) END_OF_FILE = 1 ;
    ON ERROR SNAP BEGIN ;
       ON ERROR SYSTEM ;
       PUT DATA ;
       STOP ;
    END ;
    DO WHILE( END_OF_FILE = 0 ) ;
       READ FILE( SYSIN ) INTO( DATA ) ;
       WRITE FILE( OUTFIL ) FROM( DATA ) ;
    END ;
 END READTP ;
/* VOL SHOWS OLD TAPE NAME; LABEL & DSN SHOW FILE'S POSITION ON TAPE.
//GO.SYSIN  DD DSN=FILE1,
//             DISP=OLD,
//             UNIT=TAPE,
//             LABEL=(1,NL),
//             VOL=SER=VPI17,
//             DCB=(RECFM=F,LRECL=3840,BLKSIZE=3840)
//* SYSIN REFERS TO OLD NASA TAPE; OUTFIL REFERS TO NEW SYSTEM TAPE.
//* DSN SHOWS ASSIGNED FILE NAME ON NEW TAPE; LABEL SHOWS FILE'S
//* POSITION ON TAPE; SER='SERIAL #', ASSIGNED BY SYSTEM FIRST TIME
//* AND SHOWN IN OUTPUT, SUBSEQUENTLY GIVEN BY USER.
//GO.OUTFIL DD DSN=A529E5.VPI17.FILE1,
//             DISP=(NEW,PASS),
//             UNIT=TAPE,
//             LABEL=(19,SL,EXPDT=88304),
//             VOL=(,RETAIN,,,SER=200109),
//             DCB=(RECFM=VB,LRECL=3844,BLKSIZE=16000,DEN=4)
//GO.DD1    DD DSN=&&GOSET,DISP=(OLD,DELETE)
//* THIS BLOCK MAY BE REPEATED FOR EACH FILE DESIRED WITH THE STEP #
//* CHANGED AND THE FILE ATTRIBUTES ADJUSTED.
//STEP0002 EXEC PLIXCLG
//PLI.SYSIN DD *
* PROCESS SOURCE, FLAG(I), GOSTMT ;
 READTP :
    PROCEDURE OPTIONS( MAIN ) ;
    DECLARE SYSPRINT OUTPUT PRINT FILE ;
    DECLARE SYSIN INPUT RECORD FILE ;
```

Program TAPE TAPE continued

```
TAPE     TAPE

     DECLARE OUTFIL OUTPUT RECORD FILE ;
     DECLARE DATA( 1920 ) FIXED BINARY( 15 ) ;
     DECLARE COUNT FIXED BINARY( 31 ) ;
     DECLARE END_OF_FILE FIXED BINARY( 31 ) INITIAL( 0 ) ;
     ON ENDFILE( SYSIN ) END_OF_FILE = 1 ;
     ON ERROR SNAP BEGIN ;
        ON ERROR SYSTEM ;
        PUT DATA ;
        STOP ;
     END ;
     DO WHILE( END_OF_FILE = 0 ) ;
        READ FILE( SYSIN ) INTO( DATA ) ;
        WRITE FILE( OUTFIL ) FROM( DATA ) ;
     END ;
 END READTP ;
/*
//GO.SYSIN   DD DSN=FILE2,
//              DISP=OLD,
//              UNIT=TAPE,
//              LABEL=(2,NL),
//              VOL=SER=VPI17,
//              DCB=(RECFM=F,LRECL=3840,BLKSIZE=3840)
//GO.OUTFIL DD DSN=A529E5.VPI17.FILE2,
//              DISP=(NEW,PASS),
//              UNIT=TAPE,
//              LABEL=(20,SL,EXPDT=88304),
//              VOL=(,RETAIN,,,REF=*.STEP0001.GO.OUTFIL),
//              DCB=(RECFM=VB,LRECL=3844,BLKSIZE=16000,DEN=4)
//GO.DD1     DD DSN=&&GOSET,DISP=(OLD,DELETE)
//STEP0003 EXEC PLIXCLG
//PLI.SYSIN DD *
* PROCESS SOURCE, FLAG(I), GOSTMT ;
 READTP :
     PROCEDURE OPTIONS( MAIN ) ;
     DECLARE SYSPRINT OUTPUT PRINT FILE ;
     DECLARE SYSIN INPUT RECORD FILE ;
     DECLARE OUTFIL OUTPUT RECORD FILE ;
     DECLARE DATA( 1920 ) FIXED BINARY( 15 ) ;
     DECLARE COUNT FIXED BINARY( 31 ) ;
     DECLARE END_OF_FILE FIXED BINARY( 31 ) INITIAL( 0 ) ;
     ON ENDFILE( SYSIN ) END_OF_FILE = 1 ;
     ON ERROR SNAP BEGIN ;
        ON ERROR SYSTEM ;
        PUT DATA ;
        STOP ;
     END ;
     DO WHILE( END_OF_FILE = 0 ) ;
        READ FILE( SYSIN ) INTO( DATA ) ;
        WRITE FILE( OUTFIL ) FROM( DATA ) ;
     END ;
 END READTP ;
/*
//GO.SYSIN   DD DSN=FILE3,
//              DISP=OLD,
//              UNIT=TAPE,
```

Program.TAPE TAPE continued

TAPE TAPE

```
//                LABEL=(3,NL),
//                VOL=SER=VPI17,
//                DCB=(RECFM=F,LRECL=3840,BLKSIZE=3840)
//GO.OUTFIL DD DSN=A529E5.VPI17.FILE3,
//                DISP=(NEW,PASS),
//                UNIT=TAPE,
//                LABEL=(21,SL,EXPDT=88304),
//                VOL=(,RETAIN,,,SER=200109),
//                DCB=(RECFM=VB,LRECL=3844,BLKSIZE=16000,DEN=4)
//GO.DD1     DD DSN=&&GOSET,DISP=(OLD,DELETE)
//STEP0004 EXEC PLIXCLG
//PLI.SYSIN DD *
* PROCESS SOURCE, FLAG(I), GOSTMT ;
 READTP :
    PROCEDURE OPTIONS( MAIN ) ;
    DECLARE SYSPRINT OUTPUT PRINT FILE ;
    DECLARE SYSIN INPUT RECORD FILE ;
    DECLARE OUTFIL OUTPUT RECORD FILE ;
    DECLARE DATA( 1920 ) FIXED BINARY( 15 ) ;
    DECLARE COUNT FIXED BINARY( 31 ) ;
    DECLARE END_OF_FILE FIXED BINARY( 31 ) INITIAL( 0 ) ;
    ON ENDFILE( SYSIN ) END_OF_FILE = 1 ;
    ON ERROR SNAP BEGIN ;
       ON ERROR SYSTEM ;
       PUT DATA ;
       STOP ;
    END ;
    DO WHILE( END_OF_FILE = 0 ) ;
       READ FILE( SYSIN ) INTO( DATA ) ;
       WRITE FILE( OUTFIL ) FROM( DATA ) ;
    END ;
 END READTP ;
/*
//GO.SYSIN   DD DSN=FILE4,
//                DISP=OLD,
//                UNIT=TAPE,
//                LABEL=(4,NL),
//                VOL=SER=VPI17,
//                DCB=(RECFM=F,LRECL=3840,BLKSIZE=3840)
//GO.OUTFIL DD DSN=A529E5.VPI17.FILE4,
//                DISP=(NEW,PASS),
//                UNIT=TAPE,
//                LABEL=(22,SL,EXPDT=88304),
//                VOL=(,RETAIN,,,REF=*.STEP0001.GO.OUTFIL),
//                DCB=(RECFM=VB,LRECL=3844,BLKSIZE=16000,DEN=4)
//GO.DD1     DD DSN=&&GOSET,DISP=(OLD,DELETE)
//
```

3. Program TAPE LISTE - read from the system tape and print the first 20 elements of every line.

```
TAPE     LISTE

//B0589TLO JOB 529E5,'TIM  LSEN',REGION=1536K,TIME=(,59)
/*JOBPARM LINES=40
/*PRIORITY IDLE
//STEP0001 EXEC FORTVCG
//FORT.SYSIN DD *
      INTEGER*2 DATA( 1920 )
   10 CONTINUE
         READ( 5, END = 20 ) DATA
         WRITE( 6, 99 ) ( DATA( I ), I = 1, 20 )
   99       FORMAT( ' ', 20I6 )
         GOTO  10
   20 CONTINUE
      STOP
      END
/* DSN SHOWS NAME AND LABEL SHOWS POSITION OF FILE DESIRED.
//GO.FT05F001 DD DSN=A529E5.VPI12.FILE1,
//               DISP=OLD,
//               UNIT=TAPE,
//               LABEL=(7,SL),
//               VOL=(,RETAIN,,,SER=200109) SERIAL # OF SYSTEM TAPE.
//
```

Program TAPE LISTE continued - output

File #		Counter	Day	Hour Min.	Second	Second	Channels 1	2	3	4	5	6	Once per rev	Sq. Wave	Second	Second	Channels 1	2	3	
1	1913	0	207	1059	5996	7580	-5854	-859	-1225	-5070	-651	-12975	55	604	5996	7640	-6966	-471	-1309	. . .
1	1913	1	207	1059	5997	8560	-1921	2352	-1411	-2327	938	-10687	-28	613	5997	8620	-2941	1614	-1829	(truncated)
1	1913	2	207	1059	5998	9540	-5413	535	-1495	-4682	65	-12020	55	622	5998	9600	-5856	652	-297	
1	1913	3	207	1100	0	520	-8604	1513	-103	-4922	910	-10467	-122	604	0	588	-7444	2186	306	
1	1913	4	207	1100	1	1500	-9071	-213	-1318	-4416	-38	-11266	-56	622	1	1560	-7720	147	-650	
1	1913	5	207	1100	2	2480	-9853	1061	-2673	-7046	-1385	-13481	1481	622	2	2530	-8887	193	-2812	
1	1913	6	207	1100	3	3460	-4660	839	723	-3509	37	-10145	-94	622	3	3510	-3842	867	241	
1	1913	7	207	1100	4	4440	-7132	682	-2042	-5162	-93	-9171	-12871	640	4	4490	-5634	719	-1374	
1	1913	8	207	1100	5	5420	-5781	-545	-1328	-4368	398	-8574	-103	613	5	5470	-7619	110	-1600	
1	1913	9	207	1100	6	6400	-8832	-47	18	-2771	1012	-10421	-420	613	6	6450	-7601	341	-2107	
1	1913	10	207	1100	7	7380	-5312	756	250	-3565	381	-10908	9	595	7	7430	-4770	950	-938	
1	1913	11	207	1100	8	8360	-10137	-764	-223	-4228	-911	-12755	-103	622	8	8410	-11672	-65	-2636	
1	1913	12	207	1100	9	9330	-4687	1679	853	-2246	1384	-8684	0	622	9	9390	-5386	1743	-232	
1	1913	13	207	1100	11	320	-6277	895	-2989	-5146	-642	-10007	-28	622	11	380	-5027	535	-1318	
1	1913	14	207	1100	12	1300	-7068	830	-1634	-4470	278	-9713	-47	613	12	1360	-9402	701	-520	
1	1913	15	207	1100	13	2270	-9953	-2003	-2970	-6039	-1236	-12203	-38	622	13	2330	-9595	-2132	-1328	
1	1913	16	207	1100	14	3250	-5404	719	-1829	-4100	-484	-13857	-103	622	14	3310	-4301	479	-353	
1	1913	17	207	1100	15	4220	-4706	221	27	-2928	353	-10651	18	622	15	4280	-5809	64	-539	
1	1913	18	207	1100	16	5210	-4228	1605	-715	-3694	353	-8013	9	622	16	5260	-4384	1873	371	
1	1913	19	207	1100	17	6180	-8777	802	-1244	-3389	-493	-10724	46	622	17	6240	-9834	-314	-1801	
1	1913	20	207	1100	18	7160	-7307	636	-1160	-5061	-326	-12507	-66	622	18	7220	-6240	181	779	
1	1913	21	207	1100	19	8140	-9714	-646	-2255	-7055	-2556	-14344	-56	622	19	8200	-7058	27	-715	
1	1913	22	207	1100	20	9120	-7821	147	-3026	-6233	-270	-16956	9365	6601	20	9180	-5908	-822	-1235	
1	1913	23	207	1100	22	100	-8584	-923	-2181	-5541	-1339	-11503	-75	631	22	160	-10615	-674	-1281	
1	1913	24	207	1100	23	1090	-8952	249	-3675	-6409	-3727	-16531	-5704	631	23	1140	-10330	-37	-3276	
1	1913	25	207	1100	24	2080	-9337	-1340	-1114	-5135	-307	-10458	-38	613	24	2120	-10339	-1265	-1337	
1	1913	26	207	1100	25	3060	-12508	-1947	-4269	-7027	-2110	-11119	-112	631	25	3100	-12186	-2438	-3606	
1	1913	27	207	1100	26	4030	-6369	-729	-2153	-3352	92	-8289	-38	622	26	4090	-7996	-453	-1652	
1	1913	28	207	1100	27	5010	-8793	-2003	-1875	-6021	-1060	-11753	-66	604	27	5070	-10690	-2261	-2049	
1	1913	29	207	1100	28	5980	-6902	562	-474	-2081	1105	-11928	-38	622	28	6050	-5193	221	102	
1	1913	30	207	1100	29	6960	-10495	-840	-65	-3454	241	-10926	0	622	29	7030	-9393	-563	-3759	
1	1913	31	207	1100	30	7940	-8566	-1560	-1745	-5809	157	-10862	-112	604	30	8010	-9834	-1255	2013	
1	1913	32	207	1100	31	8920	-11679	-2372	-5132	-6215	-2296	-12277	9	604	31	8980	-9549	-2141	-3666	
1	1913	33	207	1100	32	9900	-5597	-766	-93	-4176	111	-10513	-94	604	32	9960	-7022	-240	-947	
1	1913	34	207	1100	34	880	-10008	-2280	-1188	-4906	-1636	-12580	-19	613	34	940	-9365	-1357	361	
1	1913	35	207	1100	35	1860	-4945	858	-864	-3740	427	-10016	-56	622	35	1910	-7132	941	-251	

4. Program TAPE LISTL - read from the system tape and show the first few lines of the file in totality.

TAPE LISTL

```
//B0589TLO JOB 529E5,'TIM OLSEN',REGION=1536K,TIME=(,59)
/*JOBPARM LINES=40
/*PRIORITY IDLE
//STEP0001 EXEC FORTVCG
//FORT.SYSIN DD *
      INTEGER*2 DATA( 1920 )
      DO 10 I=1,3
        READ(5) DATA
        WRITE(6,99) DATA
   99   FORMAT(1H ,5I6/,191(10I6/),5I6)
   10 CONTINUE
      STOP
      END
/*
//GO.FT05F001 DD DSN=A529E5.VPI12.FILE1,
//               DISP=OLD,
//               UNIT=TAPE,
//               LABEL=(7,SL),
//               VOL=(,RETAIN,,,SER=200109)
//
```

Program TAPE LISTL continued -   output

```
          Remaining
          line length        Day    Minute

     File#          Dummy          Hour

          1    1913          0     207  1059
5996   7580 -5854   -859 -1225 -5070  -651-12975    55    604
5996   7640 -6966   -471 -1309 -4248 -1032-11726   -19    622
5996   7700 -6038   -277 -4158 -6483 -2249-13269   -28    622
5996   7750 -8271  -1006 -3573 -6972 -1887-13159   -75    613
5996   7810 -9016   -665 -2599 -5402 -1868-11781     0    613
5996   7870 -6985   -545 -1754 -5015 -1738-11909   -56    622
5996   7930 -5809   -120 -1485 -3990  -670-10255  -112    622
5996   7990 -3235   -573 -1392 -5255   -19 -9787   -19    604
5996   8050 -5266   -582   -19 -3131   381 -9539    27    622
5996   8100 -6599    184 -1513 -3731   232 -9621   -28    622
5996   8150 -5459   -139  -567 -5485   -38-11560   -66    622
5996   8210 -6075    765 -1040 -4581  -558-11376    18    631
5996   8260 -7003    793 -3016 -6344 -1422-10825    27    631
5996   8340 -6259   -231 -2293 -6529 -1078-12167    46    640
5996   8390 -5928    138  -864 -3223  -930-10320    93    622
5996   8450 -5533   1116  1011 -2272   176 -9180    27    622
5996   8500 -4779    599  1410 -3417   817-10173    18    622
  .      .      .      .      .     .      .     .      .      .
  .      .      .      .      .     .      .     .      .      .
  .      .      .      .      .     .      .     .      .      .
Time   Time Channel 2      3     4      5     6    Once  Square
(in             1                                  per    wave
seconds)                                           Rev
  .      .      .      .      .     .      .     .      .      .
  .      .      .      .      .     .      .     .      .      .
  .      .      .      .      .     .      .     .      .      .
  0      0      0      0      0 (5 zeroes indicate end of line)
```

One line of original data (1920 elements)

5. Program POS CHECK - check number of data points in each cycle.

```
POS      CHECK

//B0589TLO JOB 529E5,'T1·  'LSEN',REGION=1536K,TIME=(,59)
/*LONGKEY WINGFAL
/*JOBPARM LINES=40,CARDS=100
/*PRIORITY STANDARD
//STEP0001 EXEC FORTVCG
//FORT.SYSIN DD *
C***
C*** POS CHECK READS SYSTEM TAPE, CHECKS POSITION VS SQ WAVE.
C*** I, J, K COUNTERS; TEST & ON FOR CONTROL, POS = ROTOR
C*** ANGULAR POSITION, OPR = ONCE PER REV, SQW = SQUARE WAVE
C***
      INTEGER*2 DATA( 1920 )
      INTEGER I, J, K, POS, ON, TEST
      REAL  SQW(95500)
C***
C*** READ DATA FROM TAPE, ASSIGN SQUARE WAVE
C***
      DO 10 I=1,500
        READ( 5 ) DATA
        DO 20 J=1,191
          SQW((I-1)*191+J) = DATA(10*J+5)
   20   CONTINUE
   10 CONTINUE
C***
C*** HEADING
C***
      WRITE(6,98)
C  98 FORMAT(1H1,4('  POS SQWAVE'))
   98 FORMAT(1H1,'  K   POS   SW(K) SW(K+1)')
C***
C*** CHECK SQUARE WAVE, ASSIGN ROTOR ANGULAR POSITIONS
C*** WRITE POSITIONS AND CHANNEL VALUES
C***
      POS = 0
      ON = 0
      TEST = 0
      DO 30 K=1,95500
        IF (TEST.EQ.1) THEN
          TEST = 0
          POS = 0
          ON = 1
        ENDIF
        IF (SQW(K).GE.6000) THEN
          IF (SQW(K+1).LE.1000) THEN
            TEST = 1
            WRITE(6,92) K, (POS+1), SQW(K), SQW(K+1)
   92       FORMAT(1H ,2I6, 2F7.0)
          ENDIF
        ENDIF
        IF (ON.EQ.1) THEN
          POS = POS+1
C         WRITE(6,99) K, POS, TIME(K), CH1(K), CH2(K), CH3(K),
C    +                CH4(K), CH5(K), CH6(K), OPR(K), SQW(K)
        ENDIF
C  99   FORMAT(1H ,2I5,F8.0,8F7.0)
   30 CONTINUE
      STOP
      END
/*
//GO.FT05F001 DD DSN=A529E5.VPI10.FILE1,
//               DISP=OLD,
//               UNIT=TAPE,
//               LABEL=(1,SL),
//               VOL=(,RETAIN,,,SER=200109)
//
```

6. Program CYCLE SELECT - discard cycles where the number of points is not equal to 360.

CYCLE SELECT

```
//B0589TLO JOB 529E5,'TIM OLSEN',REGION=1536K,TIME=(,59)
/*LONGKEY WINGFAL
/*JOBPARM LINES=40,CARDS=100
/*PRIORITY STANDARD
//STEP0001 EXEC FORTVCG
//FORT.SYSIN DD *
C***
C*** CYCLE SELECT READS SYSTEM TAPE, FINDS CYCLES WITH 360 UNITS.
C*** I, J, K, L, M COUNTERS; TEST & ON FOR CONTROL, POS = ROTOR
C*** ANGULAR POSITION, OPR = ONCE PER REV, SQW = SQUARE WAVE,
C*** CYCLE = CYCLE NUMBER (IF ACCEPTED)
C***
      INTEGER*2 DATA( 1920 )
      INTEGER I, J, K, L, M, MNEW, POS, CYCLE, N
      REAL APOS(1146), CH1(1146), CH2(1146), CH3(1146)
      REAL CH4(1146), CH5(1146), CH6(1146), OPR(1146), SQW(1146)
C***
C*** READ DATA FROM TAPE, ASSIGN CHANNELS
C***
      N = 720
      DO 10 I=1,6
        READ( 5 ) DATA
        DO 20 J=1,191
          CH1((I-1)*191+J) = DATA(10*J-2)
          CH2((I-1)*191+J) = DATA(10*J-1)
          CH3((I-1)*191+J) = DATA(10*J)
          CH4((I-1)*191+J) = DATA(10*J+1)
          CH5((I-1)*191+J) = DATA(10*J+2)
          CH6((I-1)*191+J) = DATA(10*J+3)
          OPR((I-1)*191+J) = DATA(10*J+4)
          SQW((I-1)*191+J) = DATA(10*J+5)
   20   CONTINUE
   10 CONTINUE
C***
C*** HEADING
C***
      WRITE(6,98)
   98 FORMAT('1CYCLE POS ORIG APOS CHAN1  CHAN2  CHAN3 ',
     +       ' CHAN4  CHAN5  CHAN6  1/REV  SQWAVE')
C***
C*** CHECK SQUARE WAVE, ASSIGN ROTOR ANGULAR POSITIONS,
C*** CHECK CYCLE FOR 360 POSITIONS, WRITE VALUES
C***
      POS = 0
      CYCLE = 0
      DO 30 K=1,(N+360)
        POS = POS + 1
        IF (SQW(K).GE.6000) THEN
          IF (SQW(K+1).LE.1000) THEN
            IF (POS.EQ.360) THEN
              CYCLE = CYCLE + 1
              POS = 0
              DO 40 L=1,360
                M = K - 360 + L
                MNEW = (CYCLE-1)*360 + L
```

Program CYCLE SELECT continued

CYCLE SELECT

```
                 APOS(MNEW) = MNEW
                 CH1(MNEW) = CH1(M)
                 CH2(MNEW) = CH2(M)
                 CH3(MNEW) = CH3(M)
                 CH4(MNEW) = CH4(M)
                 CH5(MNEW) = CH5(M)
                 CH6(MNEW) = CH6(M)
                 OPR(MNEW) = OPR(M)
                 SQW(MNEW) = SQW(M)
                 WRITE(6,99) CYCLE, L, M, APOS(MNEW), CH1(MNEW),
     +                       CH2(MNEW), CH3(MNEW), CH4(MNEW), CH5(MNEW),
     +                       CH6(MNEW), OPR(MNEW), SQW(MNEW)
   99            FORMAT(1H ,I3,2I5,F6.0,7F7.0,F6.0)
   40          CONTINUE
             ELSE
               POS = 0
             ENDIF
           ENDIF
         ENDIF
   30 CONTINUE
      STOP
      END
/*
//GO.FT05F001 DD DSN=A529E5.VPI10.FILE1,
//               DISP=OLD,
//               UNIT=TAPE,
//               LABEL=(1,SL),
//               VOL=(,RETAIN,,,SER=200109)
//
```

7. Program VOLT PLOT - plot data from each channel (1-6), square wave, and once-per-rev signals against position

VOLT PLOT

```
//B0589TLO JOB 529E5,'TIM OLSEN',REGION=1536K,TIME=(,59)
/*LONGKEY WINGFAL
/*JOBPARM LINES=40,CARDS=100
/*PRIORITY STANDARD
//STEP0001 EXEC FORTVCGV
//FORT.SYSIN DD *
C***
C*** VOLT PLOT READS SYSTEM TAPE, PLOTS CYCLES WITH 360 UNITS.
C*** I, J, K, L, COUNTERS; N = TOTAL # OF CYCLES, POS = ROTOR
C*** ANGULAR POSITION, OPR = ONCE PER REV, SQW = SQUARE WAVE,
C*** M = ORIGINAL DATA COUNT, MNEW = DATA COUNT OF ACCEPTED CYCLES,
C*** CYCT = CYCLE COUNTER (IF ACCEPTED)
C***
      INTEGER*2 DATA( 1920 )
      INTEGER I, J, K, L, M, MNEW, POS, CYCT, N
      REAL APOS(1146), CH1(1146), CH2(1146), CH3(1146)
      REAL CH4(1146), CH5(1146), CH6(1146), OPR(1146), SQW(1146)
      N = 3
C***
C*** READ DATA FROM TAPE, ASSIGN CHANNELS
C***
      DO 10 I=1,(INT(360*(N+1)/191)+1)
        READ( 5 ) DATA
        DO 20 J=1,191
          CH1((I-1)*191+J) = DATA(10*J-2)
          CH2((I-1)*191+J) = DATA(10*J-1)
          CH3((I-1)*191+J) = DATA(10*J)
          CH4((I-1)*191+J) = DATA(10*J+1)
          CH5((I-1)*191+J) = DATA(10*J+2)
          CH6((I-1)*191+J) = DATA(10*J+3)
          OPR((I-1)*191+J) = DATA(10*J+4)
          SQW((I-1)*191+J) = DATA(10*J+5)
   20   CONTINUE
   10 CONTINUE
C***
C*** HEADING
C***
C     WRITE(6,98)
C  98 FORMAT('1CYCLE POS ORIG APOS CHAN1  CHAN2  CHAN3 ',
C    +       ' CHAN4  CHAN5  CHAN6  1/REV  SQWAVE')
C***
C*** CHECK SQUARE WAVE, ASSIGN ROTOR ANGULAR POSITIONS,
C*** CHECK CYCLE FOR 360 POSITIONS, WRITE VALUES
C***
      POS = 0
      CYCT = 0
      DO 30 K=1,((N+1)*360)
        POS = POS + 1
        IF (SQW(K).GE.6000) THEN
          IF (SQW(K+1).LE.1000) THEN
            IF (POS.EQ.360) THEN
              CYCT = CYCT + 1
              POS = 0
              DO 40 L=1,360
                M = K - 360 + L
```

Program VOLT PLOT continued

```
VOLT     PLOT

                  MNEW = (CYCT-1)*360 + L
                  APOS(MNEW) = MNEW
                  CH1(MNEW) = CH1(M)
                  CH2(MNEW) = CH2(M)
                  CH3(MNEW) = CH3(M)
                  CH4(MNEW) = CH4(M)
                  CH5(MNEW) = CH5(M)
                  CH6(MNEW) = CH6(M)
                  OPR(MNEW) = OPR(M)
                  SQW(MNEW) = SQW(M)
C                 WRITE(6,99) CYCT, L, M, APOS(MNEW), CH1(MNEW),
C    +                  CH2(MNEW), CH3(MNEW), CH4(MNEW), CH5(MNEW),
C    +                  CH6(MNEW), OPR(MNEW), SQW(MNEW)
C  99             FORMAT(1H ,I3,2I5,F6.0,7F7.0,F6.0)
   40           CONTINUE
              ELSE
                POS = 0
              ENDIF
            ENDIF
          ENDIF
   30 CONTINUE
C***
C***  PLOT SQUARE WAVE
C***
      AXX = 4.0 * N
      XFIR = 0.0
      XINC = 90.0
      AYY = 7.0
      YFIR = 0.0
      YINC = 1000.0
      APOS(N+1) = 0.0
      APOS(N+2) = 90.0
      SQW(N+1) = 0.0
      SQW(N+2) = 1000.0
      ITYPE = 0
      ITEXT = 11
      CALL PLOTS(0,0,50)
      CALL PLOT(1.,1.,-3)
      CALL VPISYM(0.75,7.0,0.12,'SQUARE WAVE',0.,11)
      CALL AXIS(0.0,0.0,'POSITION',-8,AXX,0.0,XFIR,XINC)
      CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
      CALL LINE(APOS,SQW,N,1,ITYPE,ITEXT)
      CALL PLOT(6.,6.,-999)
C***
C***  PLOT ONCE PER REV
C***
      AXX = 4.0 * N
      XFIR = 0.0
      XINC = 90.0
      AYY = 6.0
      YFIR = -15000.0
      YINC = 5000.0
      APOS(N+1) = 0.0
      APOS(N+2) = 90.0
      OPR(N+1) = -15000.0
```

Program VOLT PLOT continued

```
VOLT     PLOT

          OPR(N+2) = 5000.0
          ITYPE = 0
          ITEXT = 11
          CALL PLOTS(0,0,50)
          CALL PLOT(1.,1.,-3)
          CALL VPISYM(0.75,6.0,0.12,'ONCE PER REV',0.,12)
          CALL AXIS(0.0,0.0,'POSITION',-8,AXX,0.0,XFIR,XINC)
          CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
          CALL LINE(APOS,OPR,N,1,ITYPE,ITEXT)
          CALL PLOT(6.,6.,-999)
C***
C***      PLOT CHANNEL 1
C***
          AXX = 4.0 * N
          XFIR = 0.0
          XINC = 90.0
          AYY = 4.0
          YFIR = -16000.0
          YINC = 4000.0
          APOS(N+1) = 0.0
          APOS(N+2) = 90.0
          CH1(N+1) = -16000.0
          CH1(N+2) = 4000.0
          ITYPE = 0
          ITEXT = 11
          CALL PLOTS(0,0,50)
          CALL PLOT(1.,1.,-3)
          CALL VPISYM(0.75,6.0,0.12,'CHANNEL 1',0.,9)
          CALL AXIS(0.0,0.0,'POSITION',-8,AXX,0.0,XFIR,XINC)
          CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
          CALL LINE(APOS,CH1,N,1,ITYPE,ITEXT)
          CALL PLOT(6.,6.,-999)
C***
C***      PLOT CHANNEL 2
C***
          AXX = 4.0 * N
          XFIR = 0.0
          XINC = 90.0
          AYY = 5.0
          YFIR = -5000.0
          YINC = 2000.0
          APOS(N+1) = 0.0
          APOS(N+2) = 90.0
          CH2(N+1) = -5000.0
          CH2(N+2) = 2000.0
          ITYPE = 0
          ITEXT = 11
          CALL PLOTS(0,0,50)
          CALL PLOT(1.,1.,-3)
          CALL VPISYM(0.75,6.0,0.12,'CHANNEL 2',0.,9)
          CALL AXIS(0.0,0.0,'POSITION',-8,AXX,0.0,XFIR,XINC)
          CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
          CALL LINE(APOS,CH2,N,1,ITYPE,ITEXT)
          CALL PLOT(6.,6.,-999)
C***
```

Program VOLT PLOT continued

```
VOLT     PLOT

C***  PLOT CHANNEL 3
C***
      AXX = 4.0 * N
      XFIR = 0.0
      XINC = 90.0
      AYY = 6.0
      YFIR = -7000.0
      YINC = 2000.0
      APOS(N+1) = 0.0
      APOS(N+2) = 90.0
      CH3(N+1) = -7000.0
      CH3(N+2) = 2000.0
      ITYPE = 0
      ITEXT = 11
      CALL PLOTS(0,0,50)
      CALL PLOT(1.,1.,-3)
      CALL VPISYM(0.75,6.0,0.12,'CHANNEL 3',0.,9)
      CALL AXIS(0.0,0.0,'POSITION',-8,AXX,0.0,XFIR,XINC)
      CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
      CALL LINE(APOS,CH3,N,1,ITYPE,ITEXT)
      CALL PLOT(6.,6.,-999)
C***
C***  PLOT CHANNEL 4
C***
      AXX = 4.0 * N
      XFIR = 0.0
      XINC = 90.0
      AYY = 7.0
      YFIR = -14000.0
      YINC = 2000.0
      APOS(N+1) = 0.0
      APOS(N+2) = 90.0
      CH4(N+1) = -14000.0
      CH4(N+2) = 2000.0
      ITYPE = 0
      ITEXT = 11
      CALL PLOTS(0,0,50)
      CALL PLOT(1.,1.,-3)
      CALL VPISYM(0.75,6.0,0.12,'CHANNEL 4',0.,9)
      CALL AXIS(0.0,0.0,'POSITION',-8,AXX,0.0,XFIR,XINC)
      CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
      CALL LINE(APOS,CH4,N,1,ITYPE,ITEXT)
      CALL PLOT(6.,6.,-999)
C***
C***  PLOT CHANNEL 5
C***
      AXX = 4.0 * N
      XFIR = 0.0
      XINC = 90.0
      AYY = 5.0
      YFIR = -5000.0
      YINC = 2000.0
      APOS(N+1) = 0.0
      APOS(N+2) = 90.0
      CH5(N+1) = -5000.0
```

Program VOLT PLOT continued

```
VOLT     PLOT

      CH5(N+2) = 2000.0
      ITYPE = 0
      ITEXT = 11
      CALL PLOTS(0,0,50)
      CALL PLOT(1.,1.,-3)
      CALL VPISYM(0.75,6.0,0.12,'CHANNEL 5',0.,9)
      CALL AXIS(0.0,0.0,'POSITION',-8,AXX,0.0,XFIR,XINC)
      CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
      CALL LINE(APOS,CH5,N,1,ITYPE,ITEXT)
      CALL PLOT(6.,6.,-999)
C***
C***  PLOT CHANNEL 6
C***
      AXX = 4.0 * N
      XFIR = 0.0
      XINC = 90.0
      AYY = 6.0
      YFIR = -18000.0
      YINC = 2000.0
      APOS(N+1) = 0.0
      APOS(N+2) = 90.0
      CH6(N+1) = -18000.0
      CH6(N+2) = 2000.0
      ITYPE = 0
      ITEXT = 11
      CALL PLOTS(0,0,50)
      CALL PLOT(1.,1.,-3)
      CALL VPISYM(0.75,6.0,0.12,'CHANNEL 6',0.,9)
      CALL AXIS(0.0,0.0,'POSITION',-8,AXX,0.0,XFIR,XINC)
      CALL AXIS(0.0,0.0,'VOLTAGE',7,AYY,90.0,YFIR,YINC)
      CALL LINE(APOS,CH6,N,1,ITYPE,ITEXT)
      CALL PLOT(6.,6.,999)
      STOP
      END
/*
//GO.FT05F001 DD DSN=A529E5.VPI10.FILE1,
//               DISP=OLD,
//               UNIT=TAPE,
//               LABEL=(1,SL),
//               VOL=(,RETAIN,,,SER=200109)
//
```